Computational Algorithms and Numerical Dimensions

www.journal-cand.com

Com. Alg. Num. Dim. Vol. 4, No. 2 (2024) 145–156.

Paper Type: Original Article

Development of Upper Limb Gestures Recognition Model for Hearing and Speech Impaired Patients Using Convoluted Neural Network

Babatunde Ridwan Salam[1,*], Olusola Akinde[1], Oluwadara Joel Odeyinka[1], Samuel Enochoghene[2], Soliu Kehinde Imran[3]

[1] Department of Biomedical and Electrical, Faculty of Engineering and Technology, First Technical University, Ibadan, Nigeria; salamridwan658@gmail.com; olusola.akinde@tech-u.edu.ng; oluwadara.odeyinka@tech-u.edu.ng.
[2] Department of Electrical and Electronic Engineering, Lead City University, Off Oba Otudeko Avenue, Lagos-Ibadan Express Way Toll Gate Area, Oyo, Ibadan, Nigeria 200255; enocheghene.samuel@lcu.edu.ng.
[4] Department of Mechanical and Mechatronics, Faculty of Engineering and Technology, First Technical University, Ibadan, Nigeria; imrankehinde@gmail.com.

Citation:

Received: 01 December 2024 *Revised: 24 March 2025* *Accepted: 25 April 2025*	Salam, R., Akinde, O., Odeyinka, O. J., Enochoghene, S., & Imran, S. K. (2025). Development of upper limb gestures recognition model for hearing and speech impaired patients using convoluted neural network. *Computational algorithms and numerical dimensions*, 4(2), 145-156.

Abstract

Upper limb gesture recognition plays a major role in overcoming many difficulties and inconveniences in human life, especially for individuals with speech disabilities and hearing impairment. The ability of machines to understand human activities and process their meaning can be utilized in a vast array of applications. One of the most specific applications is sign language analysis, prediction and recognition, which will aid effective communication between healthcare providers and disabled patients. This study provides a thorough state-of-the-art technique for developing upper limb gestures and sign language recognition models, predominantly based on a computational method called Deep Learning (DL). This study implements a Convolutional Neural Network (CNN), among other DL techniques, to develop a system using different stages such as data acquisition and generation, preprocessing, classification, and model building. The model was built using different stages to analyze, detect and recognize the generated (upper limb gestures data). Performance evaluation was carried out using accuracy, precision, loss function, RMSE, MSE and MAE metrics. The CNN model evaluation results are 0.2913 Loss function, 78.73% Accuracy, 0.8409 Precision, 0.7396 Recall, 0.1127 RMSE, 0.0127 MSE and 0.0245 MAE. The study, therefore, concludes that CNN can be applied to build a gesture recognition model based on its performance. Also, insights are provided in the field of gestures and sign language recognition to facilitate future research efforts and recommend the application of reinforcement learning for developing an automated embedded system.

Keywords: CNN, Deep learning, Gesture recognition, Machine learning, Sign language, Speech disability, Upper limb gesture.

1 | Introduction

The upper limb gesture recognition field is rapidly growing in computer vision and human-computer interaction. Convolutional Neural Networks (CNNs) are widely used for gesture recognition because they

Corresponding Author: salamridwan658@gmail.com
https://doi.org/10.22105/cand.2024.491700.1161

automatically learn image features and classify gestures. 3D CNNs are valuable in medical applications, especially for physical rehabilitation using Virtual Reality (VR) therapy, where patients interact with virtual environments based on recognized upper limb gestures. According to [1], technological advances in communication engineering will help in a sustainable healthcare system, but workforce deployment may not sufficiently provide the needed capacity and speed of delivery. Information and Communications Technologies (ICT) driven by Artificial Intelligence (AI) has transformed various aspects of life, such as VR systems, communication improvement for hearing-impaired individuals, telehealth and robotics [2], [3].

Gesture recognition driven by AI has numerous applications, such as sign language translation, human-computer interaction and gaming. Despite the numerous applications, this system still faces challenges such as accuracy, robustness and flexibility [4]. Deaf and mute individuals often encounter challenges when it comes to accessing healthcare services because of communication barriers between healthcare providers and patients with hearing and speech disabilities. The shortcomings of communication barriers are ineffective diagnosis, deficient treatment, and a lack of understanding regarding the patient's needs and concerns. To alleviate this challenge, there is a need for a reliable and accurate system that can recognize and interpret upper limb gestures to facilitate effective communication. CNN mirrors the human brain's visual system and excels in image segmentation and classification tasks. These networks process images by dividing them into tiles to identify image hierarchy [5] efficiently. This work obtained defined upper limb images to develop a CNN-based gesture recognition algorithm for patients with hearing and speech disabilities.

2 | Theoretical Analysis

Various studies have explored upper limb gesture recognition and presented innovative methods to enhance human-computer interaction, assistive technology, and sports training. Authors in [6] developed a system for individuals with disabilities by converting voice input to text and gestures using two distinct systems. In [7], a gesture analysis was employed with mathematical convolution models for sign language recognition, suggesting a transition to 3D CNN for dynamic gestures. Authors in [8] combined rule learning with Bagging for posture classification, achieving interpretable models. In [9], authors introduced a system for posture recognition using RGB-D cameras and Support Vector Machines (SVMs), showing robustness. The work of authors in [10] harnessed LSTM and CNN for 3D upper limb position processing. In [11], authors utilized SVMs to classify 3D skeleton features, making them suitable for gesture recognition from body movement. Authors in [12] proposed CNN-DSCK for improved rating prediction modelling user preferences. In [5], authors employed CNN for static upper limb gesture recognition, employing image processing techniques. In dance training, [13] bone monitoring and gymnastics were merged for real-time posture correction. In [14], authors introduced a posture recognition system for dance training using Boosting algorithms. Authors in [15] developed enhanced facial expression recognition through refined training and preprocessing. In [16], authors designed a gesture-controlled smart wheelchair. Authors in [2] explored upper limb gesture recognition using camera-based image processing for human-computer interaction. In [17], authors developed a head gesture-controlled wheelchair using an accelerometer. The authors' work [18] implemented a gesture image-to-sound translation system. According to the authors' work in [19], a 3D CNN-based algorithm was introduced for recognizing drivers' gestures. In [20], authors utilized transfer learning for 3D CNNs, enhancing spatio-temporal feature learning. Authors in [21] proposed a method for gesture recognition in depth map sequences.

The literature reviewed establishes a foundation for the application of Deep Learning (DL) algorithms within the realms of Machine Learning (ML) and AI. The primary focus centres on CNNs, the most extensively utilized form of DL. The utilization of DL, particularly CNNs, has become the standard of excellence within the ML community, offering the remarkable capacity to assimilate vast datasets for learning [21]. DL has consistently surpassed traditional ML techniques, showcasing its superiority in diverse domains including cyber-security, natural language processing, bioinformatics, robotics, control, and medical information processing [20].

The existing research in upper limb gesture recognition primarily focused on general applications. There are no visible focused research outcomes addressing the specific communication needs of individuals with hearing and speech disabilities in healthcare settings. This work centres on a solution-based model using CNNs to recognize and interpret healthcare-related gestures for individuals with hearing and speech disabilities. This study focused on the detection and recognition of upper limb gestures. Image acquisition, preprocessing, enhancement, segmentation, analysis, feature extraction and pattern classification are techniques for detecting upper limb gestures. The images of certain upper limb gestures are collected, defined, and used for classification and recognition.

3 | Materials and Methods

The hand gesture recognition system is built around a computer vision application that employs ML techniques to recognize and classify human hand gestures in real time. The techniques and tools used and modes of implementing the objectives of this study are discussed in this section. The general method falls under the DL algorithm model using CNN, while the general approach, technique and tool utilized is termed the computational approach method. *Table 1* lists the major tools involved.

Table 1. List of techniques and tools used.

S/N	Techniques Deployed	Tool Used
1	Data collection	Kaggul website (existing dataset)
2	Data splitting (80;20)	Min-max scaler
3	Data preprocessing	Principal Component Analysis (PCA)
4	CNN architectural	Convolution layer Dense layer Activation functions (sigmoid)
5	Training of dataset	Multiple epoch
6	Validation of training dataset	Performances matrics
7	Testing dataset	OpenCV

3.1 | Data Generation and Model Development

Different upper limb gesture datasets used for this section[1]. The dataset consists of human hand gestures of different orientations, shapes, sizes, colours, etc., separated in content. *Table 2* describes the details of the data size split into a ratio of 80:20. 80% of the data was used for the training of the model, while 20% of the data was for the testing of the performance of the model.

Table 2. Upper limb data content.

Total Number of Folders	50
Total Number of Files	34,627
Total Size	18,813,763
Package Size	18,986,132
Ratio	100%

[1] https://www.kaggle.com/datasets/ash2703/handsignimages

3.2 | Gestures Used for Preliminary Test

Figs. 1-5 illustrate the gestures used for the preliminary test and the corresponding output command indicating the various requests of the disabled patient in which gestures used as labelled (0-4) were gestures extracted from the dataset used.

Fig. 1. (0) = emergency alert.

Fig. 2. (1) = nursing alert.

Fig. 3. (2) = hunger alert.

Fig. 4. (3) =headache alert.

Fig. 5. (4)= fever alert.

3.3 | Step-by-Step Algorithms for Building CNN Model for Upper Limb Gestures Recognition System

Achieving the objectives, which start from dataset generation for model building, is discussed in this section using several methods, such as systematic and empirical literature reviews, to generate the parameters. *Fig. 6* gives the flow diagram for the Upper Limb gesture recognition model.

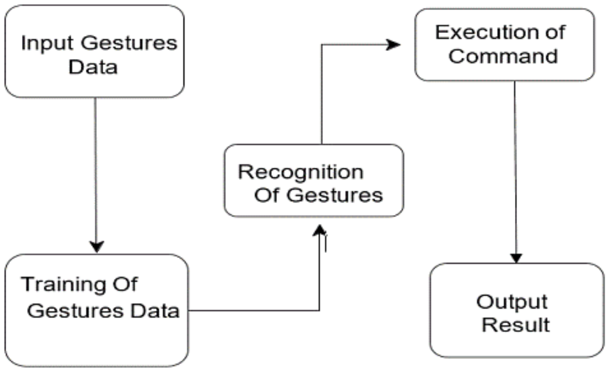

Fig. 6. Flow diagram for upper limb gesture recognition model.

Input gestures data: input gestures consist of images or sequences depicting user gestures used during training, validation, or testing phases in gesture recognition tasks.

Training the gestures data: training involves normalizing and resizing images, often augmenting them with variations, and batching the data for efficient processing in the CNN.

Recognizing the gestures: CNN's architecture has layers for convolution, pooling, and activation, which process data through the network to generate gesture recognition predictions.

Executing the command: multiple training epochs allow the CNN to improve performance, with validation steps used to avoid overfitting and ensure generalization to new data.

Output results: recognized gestures trigger specific actions or commands, enabling user interaction with systems or applications.

3.4 | Building Upper Limb Gestures Recognition Algorithms' Model Using CNN

Fig. 7 shows the sequence block diagram for building the upper limb gestures recognition algorithm model using CNN.

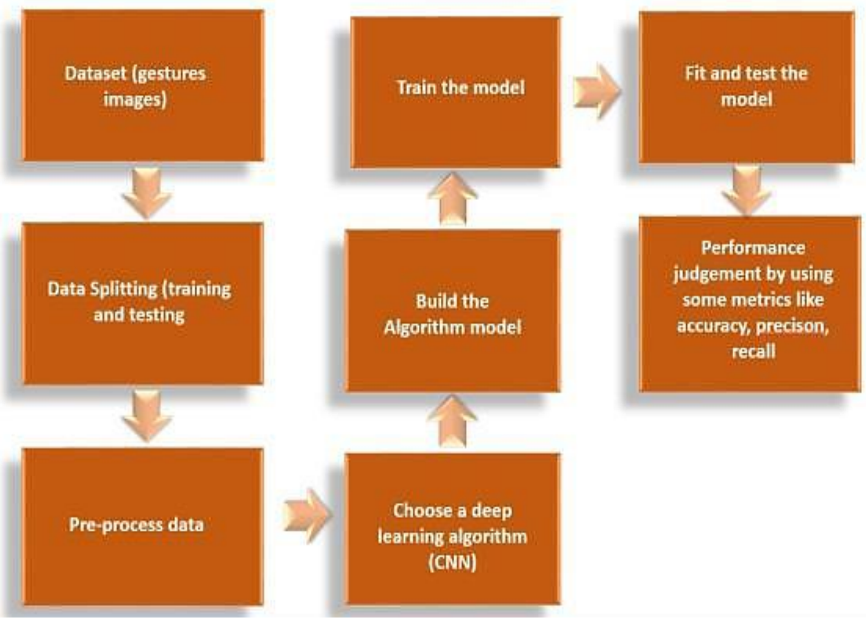

Fig. 5. Sequential block diagram of the developed model.

Dataset: the dataset contains information the model will use to classify the data. Therefore, normalization of the dataset is important. A Python library called Min-Max Scaler was imported to perform the data preprocessing. Feature scaling was used to standardize the gesture images of the features present in the data to a defined scale to normalize the gesture image. So, Feature scaling is a method used to standardize the gestures of the features present in the data to a defined gesture.

Dataset splitting: dataset splitting occurred during the model design phase, where the dataset was divided into training and testing sets with an 80:20 ratio. This facilitated the utilization of various DL hidden layers like dense, activation, batch size, sigmoid, filament, input size, and epoch to construct a CNN-DL algorithm.

Data preprocessing: further preprocessing steps were taken after dataset collection and splitting. PCA was used to extract significant features and indicators, resolve imbalanced columns and check dataset characteristics such as instance count, training and testing data count, input shape, and string presence. The processed dataset was then loaded into Jupyter in pixel format.

Building DL models from the dataset: the chosen model for the project was a CNN, a type of Artificial Neural Network (ANN) with a deep feed-forward architecture. CNNs excel in abstract feature learning, particularly for spatial data, and possess exceptional generalization capabilities. CNN encompasses processing layers that progressively learn input features, from low-level abstractions in initial layers to high-level abstractions in deeper layers. The construction involved a developed hidden layer responsible for convolution and learning, contributing to feature extraction and understanding of data [2]. The layers include:

Convolution layer

$$z^1 = h^{1-1} * W^1. \tag{1}$$

Max pooling

$$h^1_{xy} = \max_{i=0..s, j=0..s} h^{1-1}(x+i)(y+j). \tag{2}$$

Fully connected layer

$$z_l = W_l * h_{l-1}. \tag{3}$$

ReLu (rectifier)

$$\mathrm{ReLu}(z_i) = \max(0, z_i). \tag{4}$$

Soft max,

$$\mathrm{softmax}(z_i) = e^{zi} \Big/ \sum_j e^{zj}, \tag{5}$$

where W means weight, Z means activation function, and h is an arbitrary function.

4 | Results and Discussion

4.1 | Model Testing

This ensures more accurate outcomes after the model has been trained, which was done with 10 epochs (10 iterations). The model learns from the training data iteratively, adjusting its internal parameters to minimize the defined loss function. The training model involves passing batches of training data through the model, calculating the loss and updating the model's parameter. The model was trained, and the hyperparameters tuned; then, it was evaluated on the test data, which served as an unbiased measure of how well the model would generalize to unseen data. The performance of the model was validated with the following metrics:

Accuracy: one of the performance evaluation metrics in ML is accuracy. It is the fraction of predictions a model is right [3]. This is also described as the sum of True Negatives (TN) and True Positives (TP) number

divided by the sums of TP, TN, False Negatives (FN) and False Positives (FP) number. A TP or TN is an algorithm that correctly classifies data points, while a FN or FP is an algorithm that incorrectly classifies data points [4]. Mathematically,

$$\text{Accuracy} = \frac{\text{TN} + \text{TP}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}. \tag{6}$$

Precision: this is one of the ML algorithm's performance indicators; it is the quality of a positive prediction made by the algorithm. It refers to the TP number divided by the total positive prediction number, TP and FP [5]. The precision formula is given below:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{7}$$

Recall: this metric is also called the True Positive Rate (TPR). It is the data sample percentage that an ML model identifies correctly as belonging to a class of interest, which is the positive class out of the whole class samples. The formula for calculating recall is given below:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{8}$$

Root Mean Squared Error (RMSE): this is also known as root mean square deviation. It is one of the most popular metrics for quality prediction evaluation. It provides how far predictions fall from true values measured using Euclidean distance [6]. RMSE can be measured by using the formula below:

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{N} \| y(i) - \hat{y}(i) \|^2}{N}}. \tag{9}$$

N is the data point number, y(i) is the i-th measurement, and $\hat{y}$ is its corresponding prediction.

Mean Absolute Error (MAE): this metric is a very simple one which is the calculation of the absolute difference between the actual values as well as the predicted values. MAE is calculated by summing and dividing all the errors. The formula for calculating MAE is given as

$$\text{MAE} = \frac{1}{N} \sum \left| Y - \hat{Y} \right|. \tag{10}$$

N means the total number of data points, $\sum$ means the sum of the absolute value of residual, Y means actual output, and $\hat{Y}$ means predicted output.

Mean Squared Error (MSE): this metric is the finding of the squared difference between the predicted value as well as the actual value. This metric is used to prevent the cancellation of negative terms.

$$\text{MSE} = \frac{1}{n} \sum (y - \hat{y})^2 3. \tag{11}$$

$(y - \hat{y})$ is the square difference between the predicted value and the actual value.

4.2 | Developed CNN Based Gesture Recognition Algorithm

A CNN-based gesture recognition algorithm model was built using the finalized data. Anaconda, Jupyter, and libraries such as Pandas, NumPy, and Matplotlib are also available. *Fig 8* shows how the model was built.

```
    "name": "stdout",
    "output_type": "stream",
    "text": [
     "Model: \"sequential\"\n",
     "_________________________________________________________________\n",
     "Layer (type)                Output Shape              Param #   \n",
     "=================================================================\n",
     "conv2d (Conv2D)             (None, 222, 222, 32)      896       \n",
     "_________________________________________________________________\n",
     "max_pooling2d (MaxPooling2D) (None, 111, 111, 32)     0         \n",
     "_________________________________________________________________\n",
     "conv2d_1 (Conv2D)           (None, 110, 110, 64)      8256      \n",
     "_________________________________________________________________\n",
     "max_pooling2d_1 (MaxPooling2 (None, 55, 55, 64)       0         \n",
     "_________________________________________________________________\n",
     "conv2d_2 (Conv2D)           (None, 54, 54, 64)        16448     \n",
     "_________________________________________________________________\n",
     "flatten (Flatten)           (None, 186624)            0         \n",
     "_________________________________________________________________\n",
     "dense (Dense)               (None, 100)               18662500  \n",
     "_________________________________________________________________\n",
     "dense_1 (Dense)             (None, 24)                2424      \n",
     "=================================================================\n",
     "Total params: 18,690,524\n",
     "Trainable params: 18,690,524\n",
     "Non-trainable params: 0\n",
     "_________________________________________________________________\n"
    ]
   }
  ],
  "source": [
   "cnnmodel = keras.models.Sequential([\n",
   "    keras.layers.Conv2D(32, (3, 3), activation='relu', input_shape = [224, 224,3]),\n",
   "    keras.layers.MaxPooling2D(),\n",
   "    keras.layers.Conv2D(64, (2, 2), activation='relu'),\n",
   "    keras.layers.MaxPooling2D(),\n",
   "    keras.layers.Conv2D(64, (2, 2), activation='relu'),\n",
   "    keras.layers.Flatten(),\n",
```

Fig. 8. CNN model built to recognize upper limb gestures.

4.3 | Test and Validation of the Model Algorithm

CNN performance evaluation metrics results were generated using the CNN algorithm on an 80:20 splitting ratio of the dataset to train and test the model. The model algorithm gives average results in 2s per time with 78.97% accuracy, 0.8435 precision, 0.7393 recall, 0.0242 MAE, 0.0125 MSE and 0.1117 RMSE. *Table 3* presents the results of ten epochs (iteration) of the training dataset with a loss function of 0.0441. *Table 4* shows the average metrics result on the training dataset using CNN with the Splitting Ratio of 80:20 for validation extracted from the training iterations.

Table 5 shows the result (validation metrics of 20% testing dataset) of ten iterations (epochs) of validation metrics on 80:20 splitting with a loss function of 0.2913. *Table 6* shows the average validation metrics result on the testing dataset. The testing dataset (20% of the entire data set) displayed an average result in 2s per time: 78.73% accuracy, 0.8409 precision, 0.7396 recall, 0.0242 MAE, 0.0127 MSE and 0.1127 RMSE. *Fig. 9* and *Fig. 10* show the graphical representation of training and validation accuracy results and loss function results obtained from the 10 epoch iterations using an 80:20 splitting ratio on CNN, while *Table 7* represents the output of the input when each dataset was fed into the model for preliminary testing.

Table 3. CNN epoch result on 80:20 training splitting ratio of accuracy metrics (80% dataset).

Epoch Number	Accuracy (%)	Precision	Recall	MAE	MSE	Time (s)	RMSE
1	78.83	0.919	0.595	0.029	0.013	2	0.151
2	77.60	0.823	0.599	0.028	0.013	2	0.156
3	79.75	0.816	0.699	0.024	0.002	2	0.142
4	80.33	0.850	0.600	0.029	0.011	2	0.144
5	95.29	0.873	0.799	0.021	0.012	2	0.109
6	76.55	1.000	0.791	0.025	0.003	2	0.154
7	76.80	1.000	0.796	0.023	0.012	2	0.141
8	78.75	0.918	0.797	0.024	0.012	2	0.143
9	78.85	0.828	0.799	0.023	0.010	2	0.102
10	78.75	0.818	0.797	0.024	0.017	2	0.141

Table 4. Average metrics result on training dataset using CNN of the splitting ratio 80:20.

S/N	Metrics	Result
1	Loss function	0.0441
2	Accuracy	78.97 %
3	Time	2s
4	Precision	0.8435
5	Recall	0.7393
6	RMSE	0.1117
7	MSE	0.0125
8	MAE	0.0242

Table 5. CNN epoch result on 80:20 testing splitting ratio of validity metrics (20% dataset).

Epoch Number	Validity Accuracy (%)	Validity Precision	Validity Recall	Validity MAE	Validity MSE	Time (s)	Validity RMSE
1	78.82	0.910	0.495	0.039	0.017	2	0.151
2	77.62	0.824	0.599	0.038	0.015	2	0.151
3	79.71	0.815	0.699	0.024	0.025	2	0.145
4	80.31	0.854	0.600	0.029	0.018	2	0.146
5	95.29	0.878	0.799	0.024	0.019	2	0.107
6	76.55	0.8000	0.791	0.025	0.019	2	0.155
7	76.80	0.600	0.796	0.027	0.018	2	0.141
8	78.76	0.918	0.797	0.028	0.012	2	0.143
9	78.87	0.828	0.899	0.023	0.011	2	0.133
10	78.78	0.818	0.897	0.027	0.016	2	0.145

Table 6. Average metrics result on testing dataset using CNN on 80:20 splitting ratio.

S/N	Metrics	Result
1	Loss function	0.2913
2	Accuracy	78.73%
3	Time	2s
4	Precision	0.8409
5	Recall	0.7396
6	RMSE	0.1127
7	MSE	0.0127
8	MAE	0.0245

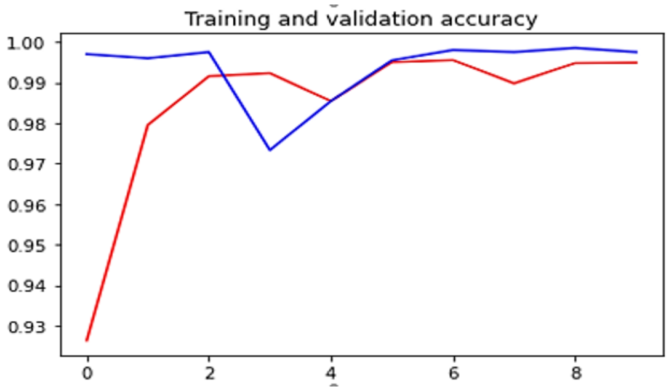

Fig. 7. Epochs training and validation accuracy.

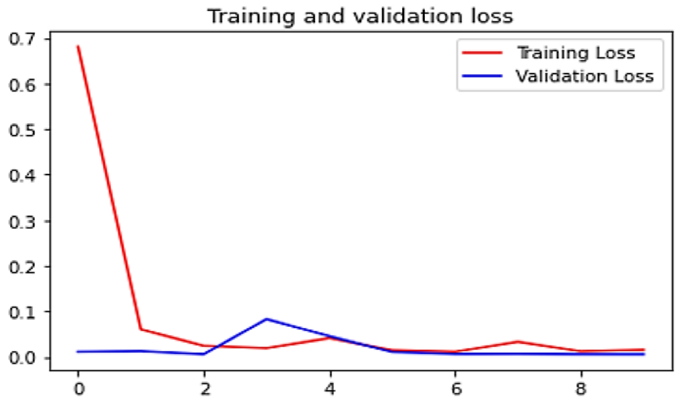

Fig. 8. Epochs training and validation loss.

Table 7. Testing result using seven (5) gestures.

Input Gestures	Output Command
0	Emergency alert
1	Nursing alert
2	Hunger alert
3	Headache alert
4	Fever alert

5 | Conclusion

The developed CNN-based algorithm model for upper limb gesture recognition achieved a 78.97% accuracy rate in recognizing healthcare-related upper limb gestures. This outcome demonstrates its potential to significantly enhance communication for individuals with hearing and speech disabilities, particularly in healthcare contexts. By carefully collecting and preprocessing diverse gesture datasets, selecting appropriate CNN architectures and leveraging techniques like transfer learning and regularization, accurate and robust models capable of recognizing complex gestures in real time can be developed. The successful implementation of such systems can lead to more inclusive and accessible technology, enabling individuals with disabilities to communicate effectively, participate in interactive experiences and access healthcare services with greater ease. As technology evolves, further research and improvements in gesture recognition systems will undoubtedly contribute to a more inclusive and interconnected world where everyone can participate and communicate effectively.

Acknowledgment

The authors would like to express sincere gratitude to all experts whose contributions have helped to improve the quality of this research work.

Author Contribution

B.R. SALAM: conceptualization, Methodology, Software, Validation, Writing – original draft and writing.

S.K IMRAN: co-writer, Conceptualization, Methodology, Software, Validation, Writing – original draft and writing.

O.K. AKINDE: Supervision, corrections – review & editing.

O.J. ODEYINKA: co-supervision, correction – review & editing.

S. O. ENOCHOGHENE: corrections, review. Lead City University, Ibadan.

Funding

This research received no specific funding from public, private, commercial or non-profit organizations.

Data Availability

The upper limb gestures dataset used for this was retrieved at https://www.kaggle.com/datasets/ash2703/handsignimages

Conflict of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

[1] Odeyinka, O. J., Ajibola, O. A., & Ndinechi, M. C. (2020). The role and trend of information and communications technology towards a pervasive healthcare system. *International journal of information communication technologies and human development*, *12*(3), 59–73. https://doi.org/10.4018/ijicthd.20200701.oa1

[2] Haria, A., Subramanian, A., Asokkumar, N., Poddar, S., & Nayak, J. S. (2017). Hand gesture recognition for human computer interaction. *Procedia computer science*, *115*, 367–374. https://doi.org/10.1016/j.procs.2017.09.092

[3] Ur Rehman, M., Ahmed, F., Khan, M. A., Tariq, U., Alfouzan, F. A., Alzahrani, N. M., & Ahmad, J. (2022). Dynamic hand gesture recognition using 3D-CNN and LSTM networks. *Computers, materials and continua*, *70*(3), 4675–4690. https://doi.org/10.32604/cmc.2022.019586

[4] Jaramillo-Yánez, A., Benalcázar, M. E., & Mena-Maldonado, E. (2020). Real-time hand gesture recognition using surface electromyography and machine learning: A systematic literature review. *Sensors*, *20*(9), 2467. https://doi.org/10.3390/s20092467

[5] Pinto, R. F., Borges, C. D. B., Almeida, A. M. A., & Paula, I. C. (2019). Static hand gesture recognition based on convolutional neural networks. *Journal of electrical and computer engineering*, *2019*(1), 4167890. https://doi.org/10.1155/2019/4167890

[6] Cooke, M., & Brown, G. J. (1999). Interactive explorations in speech and hearing. *Journal of the acoustical society of Japan (E)*, *20*(2), 89–97. https://doi.org/10.1250/ast.20.89

[7] Zhan, F. (2019). Hand gesture recognition with convolution neural networks. *2019 IEEE 20th international conference on information reuse and integration for data science (IRI)* (pp. 295–298). IEEE. https://doi.org/10.1109/IRI.2019.00054

[8] Ding, W., Hu, B., Liu, H., Wang, X., & Huang, X. (2020). Human posture recognition based on multiple features and rule learning. *International journal of machine learning and cybernetics*, *11*(11), 2529–2540. https://doi.org/10.1007/s13042-020-01138-y

[9] El Amine Elforaici, M., Chaaraoui, I., Bouachir, W., Ouakrim, Y., & Mezghani, N. (2018). Posture recognition using an rgb-d camera: exploring 3d body modeling and deep learning approaches. *2018 IEEE life sciences conference, LSC 2018* (pp. 69–72). IEEE. https://doi.org/10.1109/LSC.2018.8572079

[10] Naguri, C. R., & Bunescu, R. C. (2017). Recognition of dynamic hand gestures from 3d motion data using lstm and cnn architectures. *2017 16th IEEE international conference on machine learning and applications (ICMLA)* (pp. 1130–1133). IEEE. https://doi.org/10.1109/ICMLA.2017.00013

[11] Maret, Y., Oberson, D., & Gavrilova, M. (2018). Real-time embedded system for gesture recognition. *2018 IEEE international conference on systems, man, and cybernetics (SMC)* (pp. 30–34). IEEE. https://doi.org/10.1109/SMC.2018.00014

[12] Khan, Z. Y., & Niu, Z. (2021). CNN with depthwise separable convolutions and combined kernels for rating prediction. *Expert systems with applications*, *170*, 114528. https://doi.org/10.1016/j.eswa.2020.114528

[13] Zhou, Y. (2021). Role of human body posture recognition method based on wireless network kinect in line dance aerobics and gymnastics training. *Wireless communications and mobile computing*, *2021*(1), 9208891. https://doi.org/10.1155/2021/9208891

[14] Gao, Y., & Xu, D. (2021). Application of posture recognition service system based on information fusion smart sensor in dance training. *Journal of sensors*, *2021*(1), 4284249. https://doi.org/10.1155/2021/4284249

[15] Liu, K. C., Hsu, C. C., Wang, W. Y., & Chiang, H. H. (2019). Real-time facial expression recognition based on CNN. *2019 international conference on system science and engineering (ICSSE)* (pp. 120–123). IEEE. https://doi.org/10.1109/ICSSE.2019.8823409

[16] Noman, A. T., Khan, M. S., Islam, M. E., & Rashid, H. (2018). A new design approach for gesture controlled smart wheelchair utilizing microcontroller. *2018 international conference on innovations in science, engineering and technology (ICISET)* (pp. 64-68). IEEE. https://doi.org/10.1109/ICISET.2018.8745607

[17] Nasif, S., & Khan, M. A. G. (2017). Wireless head gesture controlled wheel chair for disable persons. *2017 IEEE region 10 humanitarian technology conference (R10-HTC)* (pp. 156–161). IEEE. https://doi.org/10.1109/R10-HTC.2017.8288928

[18] Bhuvaneshwari, C., & Manjunathan, A. (2020). Advanced gesture recognition system using long-term recurrent convolution network. *Materials today: proceedings*, *21*, 731–733. https://doi.org/10.1016/j.matpr.2019.06.748

[19] Molchanov, P., Gupta, S., Kim, K., & Kautz, J. (2015). *Hand gesture recognition with 3d convolutional neural networks* [presentation]. Proceedings of the IEEE conference on computer vision and pattern recognition workshops (pp. 1–7). https://B2n.ir/nj7304

[20] Al-Hammadi, M., Muhammad, G., Abdul, W., Alsulaiman, M., & Hossain, M. S. (2019). Hand gesture recognition using 3D-CNN model. *IEEE consumer electronics magazine*, *9*(1), 95–101. https://doi.org/10.1109/MCE.2019.2941464

[21] Azad, R., Asadi-Aghbolaghi, M., Kasaei, S., & Escalera, S. (2019). Dynamic 3D hand gesture recognition by learning weighted depth motion maps. *IEEE transactions on circuits and systems for video technology*, *29*(6), 1729–1740. https://doi.org/10.1109/TCSVT.2018.2855416

[22] Ghosh, A., Sufian, A., Sultana, F., Chakrabarti, A., & De, D. (2020). Fundamental concepts of convolutional neural network. *Recent trends and advances in artificial intelligence and Internet of Things*, 519-567. https://doi.org/10.1007/978-3-030-32644-9_36